ITEC116 – Prof. Vincent Dallego

LESSON 1: What is an API?

API's (Application Programming Interface)

- These are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols.
- Refers to any software with distinct functions.
- Defines the rules that you must follow to communicate with other software systems.

A Web API as a gateway between clients and resources on the WEB

- **Clients –** Users who want to access information from the web.
- **Resources –** Are the information that different applications provide to their clients. It can be images, videos, text, number or any type of data.

4 DIFFERENT WAYS THAT API'S CAN WORK

1. **SOAP API's** (Simple Object Access Protocol)
    - Client and server exchange messages using XML. This is less flexible API that was more popular in the past.
2. **RPC API's** (Remote Procedure Calls)
    - The client completes a function or a procedure on the server and the server sends the output back to the client.
3. **WEBSOCKET API's**
    - Is another modern web API that uses JSON objects to pass data.
    - Supports two ways of communication between clients apps and the server.
4. **REST** (Representational State Transfer)
    - Most popular and flexible API's found on the web today.
    - Is a software architecture that imposes conditions on how an API should work.

2 PRINCIPLES OF THE REST Architectural Style

- ❖ **Uniform Interface –** is fundamental to the design of any RESTFUL Web Service. It indicates that the server transfers information in a standard format.
- ❖ **Uniform Interface imposes four architectural constraints**

4 BENEFITS OF REST API's

1. **Integration –** are used to integrate new applications with existing software systems. This increases development speed because each functionality doesn't have to be written from scratch.
2. **Innovation –** entire industries can change with arrival of new app.
3. **Expansion –** present a unique opportunity for businesses to meet their clients' needs across different platforms
4. **Ease of Maintenance -** it acts a gateway between two systems.

4 TYPES OF API's

1. **Private API's**
   - Internal to an enterprise and only used for connecting systems and data within the business.
2. **Public API's**
   - Open to the public and may be used for anyone.
3. **Partner API's**
   - Accessible to authorized external developers to aid business to business partnerships.
4. **Composite API's**
   - Combine two or more different API's to address complex system requirements or behaviors.

5 STEPS REQUIRED FOR HIGH-QUALITY API DESIGN

1. **PLAN THE API –** Provide the blueprint for your API design.
2. **BUILD THE API –** API designers prototype API's using boilerplate code.
3. **TEST THE API –** Must be done to prevent bugs and defects.
4. **DOCUMENT THE API –** Acts as guide to improve usability.
5. **MARKET THE API –** Listing API can allow you to monetize it.

LESSON 2: What is RESTFUL API?

RESTFUL API – is an interface that two computer systems use to exchange information securely over the internet.

3 BENEFITS OF A RESTFUL API

1. Scalability – **can scale efficiently because REST optimizes client-server interactions.**
2. Flexibility – **support total client server separation.**
3. Independence – **are independent of the technology used.**

5 RESTFUL API CLIENT REQUEST CONTAIN

1. **Unique Resource Identifier –** the server identifies each resource with unique identifiers. For REST services the server typically performs resource identification by using a *Uniform Resource Locator URL*
2. **Method –** it implements using the *Hypertext Transfer Protocol HTTP* method that tells the server what it needs to do to resources.
   - **HTTP Methods consist of:**
     - **GET –** access resources that are located at the specified URL on the server.

        - **POST –** it sends data to the server.
        - **PUT –** update existing resources on the server.
        - **DELETE –** request to remove resource. It can change the server state.

3. **HTTP Header –** is the metadata exchanged between the client and server.
4. **Data –** request might include data for the POST, PUT, and other HTTP methods to work successfully.
5. **Parameter –** can include parameters that give the server more details about what needs to be done.
    - **3 TYPES OF PARAMETERS**
        - **Path Parameter –** specify URL details.
        - **Query Parameter –** that request more information about the resource.
        - **Cookie Parameter –** that authenticate clients quickly.

RESTFUL Authentication Methods – a web service must authenticate requests before it can send a response.

5 COMMON AUTHENTICATION METHODS

1. **HTTP AUTHENTICATION**
    - Defines some authentication schemes that you can use directly when you are implementing REST API.
2. **BASIC AUTHENTICATION**
    - The client sends the username and password in the request header.
3. **BEARER AUTHENTICATION**
    - It refers to the process of giving access control to the token bearer.
4. **API KEYS**
    - The server assigned a unique generated value to a first-time client.
5. **OAUTH**
    - Combines passwords and tokens for highly secure login access to any system.

RESTFUL API server response contain MAIN Components

1. **Status Line** – contains three-digit status code that communicate a success of failure.
   - **200:** success
   - **201:** post method success
   - **400:** incorrect request
   - **404:** resource not found
2. **Message Body –** contains the resource representation. Format can be in XML or JSON formats.
3. **Header –** contains headers or metadata about the response. It gives more context about the response.

LESSON 3: System Integration

System Integration – is the act of taking many disparate systems and workflows and bringing them together into a single system that operates more effectively.

Advantages of System Integration

- **Improve System Security –** help IT teams and business leaders keep an eye on what data is visible by which individuals or groups.
- **Cost and Storage Savings –** remove redundancy in applications and data.
- **Real Time Data –** data can be updated in real time.
- **Better Analysis –** also solve the age-old problem of one hand not knowing what the other is doing.
- **Accelerated Growth and Innovation –** due to improve efficiency and effectiveness.

3 TYPES OF SYSTEM INTEGRATION

1. **Vertical Integration –** can be thought of top to bottom or start to finish.
2. **Horizontal Integration –** can be thought of as left to right or across
3. **Star Integration –** is less common and typically born out of an inability to service horizontal or vertical integration.

```
@GetMapping("/CheckPalindrome/{number}")
public String CheckPalindrome(@PathVariable(value = "number")
int number) {
   String string = String.valueOf(number);
   String output = new
StringBuilder(string).reverse().toString();
   if (output.equals(string))
       return "The " + number + " is Palindrome";
   else
       return "The " + number + " is not Palindrome";
```

```
package com.example.itec116_springbooth;

public class SimpleCalculator {

   public String getOperator() {
       return operator;
   }

   public void setOperator(String operator) {
       this.operator = operator;
   }

   public double sum(){ return firstNumber+secondNumber;
   }
   public double diff(){
       return firstNumber-secondNumber;
   }
   public double product(){
       return firstNumber*secondNumber;
   }
   public double quotient(){
       return firstNumber/secondNumber;
   }

   public double getFirstNumber() {
       return firstNumber;
   }

   public void setFirstNumber(double firstNumber) {
       this.firstNumber = firstNumber;
   }

   public double getSecondNumber() {
       return secondNumber;
   }

   public void setSecondNumber(double secondNumber) {
       this.secondNumber = secondNumber;
   }

   private double firstNumber;
   private double secondNumber;
   private String operator;


}
```

```
@PostMapping("/SimpleCalculator")
public String SimpleCalculator(@RequestBody SimpleCalculator
simpleCalculator) {
   switch (simpleCalculator.getOperator()) {

       case "+":
           return "The sum of number is " +
simpleCalculator.sum();
       case "-":
           return "The difference of number is " +
simpleCalculator.diff();
       case "*":
           return "The product of number is " +
simpleCalculator.product();
       case "/":
           return "The quotient of number is " +
simpleCalculator.quotient();
       default:
           return "The number is invalid!";
```

```
@GetMapping("/CheckOddEven/{number}")
public String CheckOddEven(@PathVariable(value = "number") int
number) {
   if (number % 2 == 0) {
       return "The number is " + number + " an even number";
   } else {
       return "The number is " + number + " an odd number";
   }


}
```